**Classes and Objects**

A class is a template for objects, and an object is an instance of class.

In a class, variables are called properties and functions are called methods!

Objects of a class are created using the new keyword.

Class is a programmer-defined data type.

**Property**

ক্লাস এর মেম্বার ভ্যারিয়েবল।

**Member function and veriable**

ক্লাসের ভিতর ভেরিয়েবল কে বলা হয় প্রোপর্টিজ বা মেম্বার ভেরিয়েবল এবং ফাংশনকে বলা হয় মেথড বা মেম্বার ফাংশন বা (behaviour)বিহেবিয়ার।

**Constructor**

পিএইচপির বিশেষ ধরণের বিল্ট-ইন মেথড। অবজেক্ট তৈরির সময় প্রোপার্টির ভ্যালু এসাইনে সম্মতিদেয়। অবজেক্ট তৈরি হলে এই মেথডটি স্বয়ং ক্রিয়ভাবেই সম্পাদিতহয়।দুটি আন্ডারস্কোর(\_\_) দিয়ে construct মেথড শুরু হয়।

**Function** এবং **Constructor** এরপার্থক্যগুলো

· Function হল Group of Code যা একটি নির্দিষ্ট কাজ করে অন্যদিকে Constructor এর কাজ হল class এর Instance তৈরি করা।

Function এর return type থাকে কিন্তু Constructor এর কোন return type নেই এমনকি void ও ব্যবহার করা যায়না।

Function এ final, static, abstract এই ধরনের Modifier গুলো ব্যবহার হয় কিন্তু Constructor এ এই ধরনের Modifier গুলো ব্যবহার করা যায় না।

Function এ this, super কিওয়ার্ড ব্যবহার হয় Working Class এবং Parent Class এর Instance কে Refer করার জন্য অন্যদিকে constructor এ this, super কিওয়ার্ড ব্যবহার হয় Working Class এবং Parent Class এর Constructor কে Call করারজন্য।

**PHP: Overloading and Overriding –**

**অভারলোডিং** তখনই ঘটে যখন একই ক্লাসে দুই বা তার অধিক মেথড এর নাম একই কিন্তু প্যারামিটার ভিন্ন হয়।

**Overriding** এর অর্থ হলো দুটি মেথডের নাম এবং প্যারামিটার (method signature) একই। একটি মেথড থাকে প্যারেন্ট ক্লাসে এবং অপরটি থাকে চাইল্ড ক্লাসে।

Main parts of oop :- (AEIP)

1. **Abstraction**:-

An abstract class is a class that contains at least one abstract method. An abstract method is a method that is declared, but not implemented in the code.

An abstract class or method is defined with the abstract keyword: **Abstract Class কি?**

Abstract Class এমন এক ধরনের ক্লাস যার অবজেক্ট সৃষ্টি করা যায় না।অন্য ক্লাস(subclass) থেকে extend করাই হল এর উদ্দেশ্য।

· **Abstract Method কি?**

Abstract method হল এমন এক ধরনের মেথড যার শুধু ঘোষণা (declaration) থাকে,বাস্তবায়ন(implementation) থাকেনা।

**Abstract Method কেন ব্যবহার করা হয়?**

ধরি, আমরা একটি মানুষের আচরনের ক্লাস তৈরি করতে চাই, যার Base Class শুরু হবে মানুষ নাম দিয়ে। বিভিন্ন মানুষের বিভিন্ন রকম আচরন থাকতে পারে যেমন কেউ গান পছন্দ করে, কেউ নাচ পছন্দ করে, কেউ ক্রিকেট খেলা পছন্দ করে কিন্তু সব মানুষের কমন কিছু

বৈশিষ্ট্য আছে যেমন সব মানুষেই খাবার গ্রহন করে, সব মানুষেই ঘুমায়।যখন একটি বেস ক্লাসে ভিন্ন ভিন্ন বৈশিষ্ট্যর মেথড থাকে,তখন ওই ক্লাসকে abstract class হিসেবে ঘোষণা করা হয়।

**Interface :**

ইন্টার ফেসের মধ্যে কোন property থাকতে পারবে না। Interface এর method অবশ্যই public হতেহবে।তবে Interface এরসকল method ই abstract এরমত। function এর মধ্যে কোন implement করা যায় না। Interface কে একটি class implement করতেহলে implement keyword ব্যবহারকরতেহয়।

· আমরা একটি চাইল্ড ক্লাসে অসংখ্য ইন্টারফেস সংযুক্ত করতে পারি।একটি ক্লাসে একাধিক ইন্টারফেস সংযুক্ত করার জন্য কমা(,) চিহ্ন ব্যবহার করে ইন্টারফেস সমূহক পৃথক করা হয়।

· Interface class কে অ্ন্য ক্লাস implement করতে হলে তার সকল মেথড কে implement কেরতে হয়।

· ইন্টারফেসে ক্লাসের সকল মেথড অবশ্যই public হতে হবে, কিন্তু অ্যাবস্ট্রাক্ট ক্লাস সকল অ্যাক্সেস মডিফায়ার সমর্থন করে।

· একটি চাইল্ড ক্লাস এক সাথে একাধিক ইন্টারফেস ব্যবহার করতে পারে, কিন্তু অ্যাবস্ট্রাক্ট ক্লাসের ক্ষেত্রে চাইল্ড ক্লাস একের অধিক অ্যাবস্ট্রাক্ট ক্লাস থেকে ইনহেরিট হতে পারেনা।

\*\*ইন্টারফেস ব্যবহারের আরকেটি সুবিধা হচ্ছে যখন একসাথে অনেক প্রোগ্রামার কাজ করবে তখন হয়ত প্রজেক্ট ম্যানেজার একটা ইন্টারফেস তৈরী করে জুনিয়র প্রোগ্রামার কে দিয়ে দেবে।এতে জুনিয়র প্রোগ্রামারকে আর এপ্লিকেশন নিয়ে ভাবতে হবেনা,সে শুধু ইন্টারফেস দেখে দেখে তার ক্লাসগুলিতে মেথডগুলি পূর্ন (মেথডে কোড লিখবে) করবে।

| ইন্টারফেসএবংঅ্যাবস্ট্রাক্টক্লাসেরমধ্যেপার্থক্য | | |
| --- | --- | --- |
|  | **ইন্টারফেস** | **অ্যাবস্ট্রাক্টক্লাস** |
| কোড | অ্যাবস্ট্রাক্টমেথড, কনস্ট্যান্ট | অ্যাবস্ট্রাক্টমেথড, কনস্ট্যান্ট, কনক্রিটমেথড, কনক্রিটভ্যারিয়েবল |
| অ্যাক্সেসমডিফায়ার | public | public  private  protected |
| প্যারেন্টক্লাসেরসংখ্যা | একটিচাইল্ডক্লাসেরএকাধিকপ্যারেন্টক্লাসবাইন্টারফেসথাকতেপারে। | একটিচাইল্ডক্লাসেরশুধুমাত্রএকটিপ্যারেন্টক্লাসবাঅ্যাবস্ট্রাক্টথাকবে। |

**Syntax of interface :**

Interface interfaceName{

body—----

}

Interface car{

Public function sound();

}

## **Traits :PHP তে multiple inheritance সাপোর্টকরেনা। এইসমস্যা solve করারজন্য Traits use করাহয়। Traits এরকোন object/instance create করাযায়না। একটি class multiple traits কে use করতেপারে।**

**Syntax of traits:**

Traits traitsName{

body —--------------

}

Traits message{

Public function show(){

—---------

}

}

How to use traits—

Class এরমধ্যে traits কে use করারজন্য use keyword ব্যবহারকরাহয়।

Example– use traitsName

Class a {

Use message;

}

Inheritance

Inheritance অর্থ হলো উত্তরাধিকার । যখন একটি class(parent) এর বৈশিষ্ট্য (methods & property) inherit করে অন্য class(child) এ ব্যবহার করা হয় তাকে Inheritance বলে । Inherit করা জন্য extends keyword ব্যবহার করে ।

* Child class তার Parent class এর সকল public, protected মেথড এবং প্রপারটি ব্যবহার(inherit) করতে পারবে। Inherit করা জন্য extends keyword ব্যবহার করতে হয়।

Types of inheritance in php

PHP তে তিন ধরণের inheritance রয়েছে। Single, multiple and multilevel inheritance.

* Single inheritance: যখন একটি child class শুধুমাত্র একটি parent class কে inherit করে তাকে Single Inheritance বলে।

Example:-

**<?php**

**class ParentClass**

**{**

**function message()**

**{**

**echo "hi";**

**}**

**}**

**class childClass extends ParentClass**

**{**

**function display()**

**{**

**echo "bye";**

**}**

**}**

**$Obj = new childClass ();**

**$Obj->message();**

**$Obj->show();**

**?>**

* Multiple Inheritance: যখন একটি child class একাধিক parent class কে inherit করে তাকে Multiple Inheritance বলে ।

PHP তে multiple inheritance সাপোর্ট করে না । Multiple inheritance এর সীমাবদ্ধতা দূর করার trait ব্যবহার করা হয়

* Multilevel Inheritance: ধরি A, B, C তিন টা ক্লাস , যদি B ক্লাস A কে এবং C ক্লাস B কে inherit করে তাকে Multilevel Inheritance বলে ।

## PHP - Overriding Inherited Methods

# Namespaces

নেমস্পেস মুলত ডিরেক্টরি বা ফাইল নির্দেশক। Same নামের ক্লাস বা ফাংশনকে বিভিন্ন namespaces এ define করা যায় এবং namespaces ব্যবহার করে element গুলো অ্যাক্সেস করা যায় ।

আপানার লারাভেল প্যাকেজে বা অন্য যেকোনো পি এইচ পি ডেভেলপমেন্ট প্রোজেক্ট এ যদি একই নামে একাধিক function , class , interface , constant , trait , abstract থাকে তখন সমস্যা সৃষ্টি হবে । কোন টা ব্য্যবহার হচ্ছে সেটা ফিক্স করা সম্ভব না । এই সমস্যা সমাধানের জন্য নেমস্পেস ব্যবহার করা হয় ।

নেমস্পেস কে প্রকাশ করা হয় namespace দিয়ে । পিএইচপি (PHP) / লারাভেল (Laravel) এর সকল ফাইল এর উপরে (top) এ  
নেমস্পেস (namespace) ব্যবহার করতে হয় । উদাহরণ namespace App\Http\Controllers; সাব-নেমস্পেস ব্যাকস্লাশ (\) দিয়ে পৃথক করা হয় ।

laravel এর একটি ফাইল একটা নেমস্পেস হবে । বাকি গুলা use ব্যবহার করে নেমস্পেশ দেখাতে হবে ।

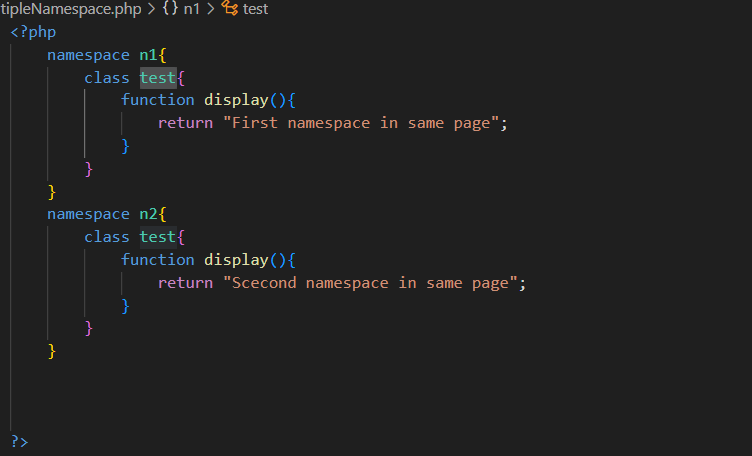
* নেমস্পেস ব্যবহারের সুবিধা হচ্ছে পুনরায় অন্য ফাইল এ ব্যবহার করা যায় ।

### **নেমস্পেস ইমপোর্ট (importing/aliasing)**

নেমস্পেসের মাধ্যমে ইমপোর্ট করা হয় use operator দিয়ে । লারাভেল প্যাকেজ এর ডিফল্ট Controller এর কোড ।

* aliasing / উপনাম/ডাক-নাম - সবার নাম অনেক বড় হয় , কিন্তু সহজে ডাকার জন্য ছোট একটা নাম থাকে , ডাক নাম । এই ছোট বা ডাক নাম কে alias বলা হয় ।

as দিয়ে aliasing নেমস্পেস ইমপোর্ট করা হয় ।



## 

## 

## 

## Static Properties and Method

Class এর মধ্যে যে সকল method, property কে object create ছাড়া সরাসরি class দিয়ে access করার জন্য static ব্যবহার করা হয় ।

Static method, static property এবং constant কে class এর ভিতরে অথবা বাহিরে ব্যবহার করার জন্য Scope resolution operator(::) ব্যবহার করা হয় ।

Static method শুধু মাত্র static property বা constant কে করতে call করতে পারবে ।

কোন ক্লাসে স্টাটিক মেথড বা প্রোপার্টিজ এর প্রয়োজন হলে একসেস মডিফায়ার (public, private, protected কিওয়ার্ড) এর পর static শব্দটি দিতে হবে।

\*স্টাটিক মেথডে শুধুমাত্র স্টাটিক ভেরিয়েবল একসেস নেয়া সম্ভব,কোন ভেরিয়েবল যদি স্টাটিক না হয় তাহলে $this সিডো ভেরিয়েবল ব্যবহার করে একসেস নেয়া সম্ভব নয় কারন এই ক্লাসের instantiation করা হচ্ছেনা।

\*\*স্টাটিক মেথড বা প্রোপার্টিজ স্ক্রিপ্টে গ্লোবাল ভেরিয়েবলের মত,যেকোন জায়গা থেকে একসেস নেয়া যায় (ক্লাসের নাম তারপর ডাবল ক্লোন চিহ্ন দিয়ে)

\*\*ক্লাসের ভিতরে একসেস নিতে self তারপর ডাবল ক্লোন চিহ্ন আর ক্লাসের বাইরে হলে className:: এভাবে (উপরের নিয়মটির মত)

Polymorphism

Polymorphism – বহুরূপি । অর্থাৎ একই অবজেক্ট বহুরূপে ব্যবহার করা যায় । দুটি ধরণের পলিমরফিজম রয়েছে

\*\*মেথড ওভাররাইডিং পলিমরফিজমের উৎকৃষ্ট উদাহরন।

একই মেথড বিভিন্ন ক্লাসে বিভিন্ন কাজ করবে প্যারামিটার ইত্যাদির উপর ভিত্তি করে।

Polymorphism - বিভিন্ন ক্লাসের যেসব মেথডসমূহ একই কাজে ব্যবহৃত হয় তাদের নামকরণ একই হওয়া।

## Encapsulation

এনক্যাপসুলেশন এমন একটা পদ্ধতি যেটা দ্বারা স্ক্রিপ্টে প্রোপার্টিজ,মেথড ইত্যাদির স্কোপ ঠিক করে দেয়া যায় (আবার এটা মনে করবেননা যে,মেথড/প্রোপার্টিজকে একসেস মডিফায়ার দিয়ে সীমানা ঠিক করে দেওয়া শুধুমাত্র এটাই এনক্যাসুলেশন।বরং অবজেক্টে ডেটাগুলি বিভিন্নভাবে মোড়ানো থাকার কারনে যে সুরক্ষিত, এটাই এনক্যাপসুলেশন)।

**একসেস মডিফায়ার** (public, private এবং protected) দিয়ে এই কাজ করা যায়।যখনি এই কিওয়ার্ডগুলি দিয়ে কোন মেথড বা প্রোপার্টিজ এর স্কোপ ঠিক করা হবে তখনি এটা নির্দিষ্ট হয়ে যাবে যে এই মেথড এবং প্রোপার্টিজ স্ক্রিপ্টের কোথায় ব্যবহার করা যাবে আর কোথায় যাবেনা। ধারনা আসছেনা?অর্থ্যাৎ এই সুবিধা শুধু কোডাররা পাবে যখন নাকি একই প্রজেক্টে অনেক প্রোগ্রামার কাজ করবেন।

Traits

PHP তে multiple inheritance সাপোর্ট করে না । এই সীমাবদ্ধতা দূর করার trait ব্যবহার করা হয় । Trait user করে multiple inheritance করা যায় . যা PHP 5.4 এ প্রথম ব্যবহার করা হয়।

Traits অনেকটা class এর মতোই, Trait কে define করা হয় ক্লাসের মত করেই trait কিওয়ার্ডটি ব্যবহার করে। তবে class এর মত object তৈরী করা যায় না। কিন্তু একাধিক trait এর property এবং method গুলোকে একটি single class এর মধ্যে ব্যবহার করা যায়।

Note: যদি trait, parent class এবং child class এ একই নামের function পাওয়া থাকে , তখন প্রথমে child`, তারপর trait, তারপর parent class এর method টি execute হবে

Magic Method

ম্যাজিক মেথডগুলি ডাবল আন্ডারস্কোর (\_\_) দিয়ে শুরু হয় এবং সবগুলির একটা কমন বৈশিষ্ট্য আছে তাহল এই মেথডগুলি কখনও কল করতে হয়না বরং অটোমেটিক এক্সিকিউট হয় ।

\_\_construct(), \_\_destruct(), \_\_call(), \_\_callStatic(), \_\_get(), \_\_set(), \_\_isset(), \_\_unset(), \_\_sleep(), \_\_wakeup(), \_\_toString(), \_\_invoke(), \_\_set\_state() এবং \_\_clone() ফাংশনকে পিএইচপি ম্যাজিক মেথড বলা হয়।